Health Monitoring of Federated Future Internet Experimentation Facilities

Thijs Walcarius, Wim Vandenberghe,
Brecht Vermeulen, Piet Demeester
IBCN, INTEC
Ghent University – iMinds
Ghent, Belgium
firstname.lastname@intec.ugent.be

Dai Davies
DANTE
City House, Hills Road
Cambridge CB2 1PQ, United Kingdom
dai@dante.net

***Abstract*—The federation of Future Internet testbeds as envisaged by the Fed4FIRE project is a complex undertaking. It combines a large number of existing, independent testbeds in a single federation, and presents them to the experimenter as if it were a single infrastructure. Operating and using such an infrastructure requires a profound knowledge of the status of the health of the underlying independent systems. Inspired by network monitoring techniques used to operate the Internet today, this paper considers how a centralized health monitoring system can be set up in a federated environment of Future Internet Experimentation Facilities. We show why it is a vital tool for experimenters and First Level Support in the federation, which health monitoring information must be captured, and how this information can be displayed most appropriately.**

Keywords—Future Internet Experimentation Facilities, FIRE, Fed4FIRE, health monitoring, federation, support, operations

I. INTRODUCTION

A federation of Future Internet testbeds as envisaged by the Fed4FIRE project [1] is a complex undertaking [2]. It combines a large number of existing, independent testbeds in a single federation (see figure 1), and presents them to the experimenter as if it were a single infrastructure. But in reality no single entity owns or operates this entire federated testbed. Every testbed remains autonomous, deciding independently on its operational aspects such as maintenance, support, fault management, etc. You could compare this with how the Internet works. The Internet can be seen as a federation of independent networks (called autonomous systems), which decided to collaborate to create an enlarged and more performant network that they can offer to their local user base. An example of this is the pan-European research network where every country provides its own autonomous system (called a National Research and Education Network or NREN), which are interconnected with each other through the GÉANT network. To manage this complex federated network, both the NRENs keep track of the health status of their own network, but on the federated level additional monitoring is done by DANTE, which is responsible for the operation of the GÉANT network [3]. Given the clear resemblance between a federated network as exemplified by the Internet and the federation of Future Internet testbeds as envisaged by the Fed4FIRE project, adopting one of the Internet's operational procedures within the federated testbed domain seems very sensible. As a result, health monitoring information should be made available both by the testbeds and the federation-level components. This requires some changes in the operational approach of these stakeholders. Today, testbed-specific operational procedures (including health monitoring, fault management and experimenter support) are typically quite informal, but are adequate in the context of providing the testbed to its local userbase. However, these mechanisms will not scale up when looking at federated operational models. For instance, when assessing if the federated infrastructure is up and running, it is not feasible to check the individual health information available at every testbed, and deduce the overall health status of the federated testbed manually.

First Level Support (FLS) is the function responsible for maintaining the operation of the federation. FLS operators and other staff in charge of keeping the federation operational have to be able to respond quickly to any service interruption. They should therefore be able to retrieve overall health monitoring information in a very efficient manner. In a similar way, when experimenters face issues with experiments that unexpectedly malfunction, it is not straightforward for them to assess manually if the faults are related to problems with their specific experiment, or with the testbeds where it is deployed on. If the federation can provide an 'easy to understand' overview of the federation's health status, this would allow the experimenter to assess if there is anything wrong with one or more of the testbeds belonging to the experiment. This would enable the experimenter to take preventative measures immediately (move the experiment to similar testbeds which are running smoothly, change the experiment design, postpone the experiment until the issues with the desired testbeds are resolved, etc.). However, at the moment it is unclear how such an overview of the federation's health status can be realized. It is also unknown if such a system would be as useful in practice as it is in theory. Finding out the answers to these questions is exactly the scope of this paper.

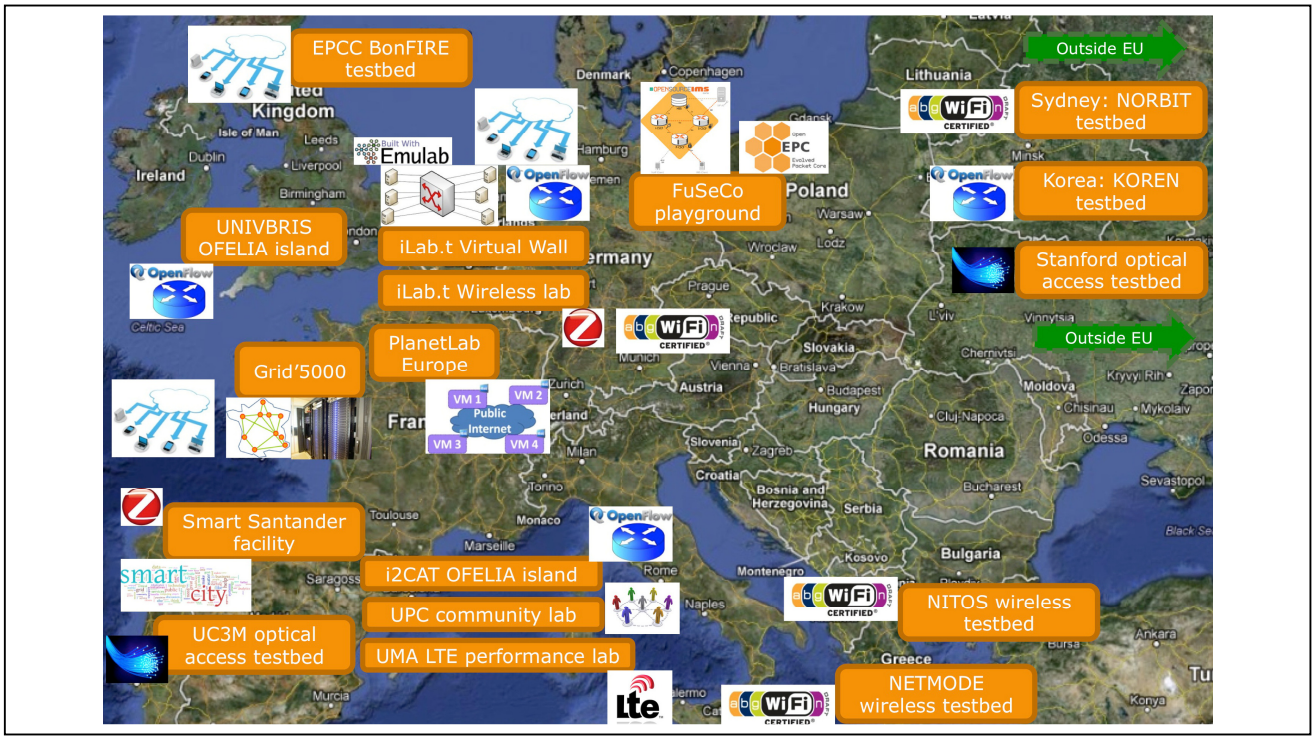


Fig. 1. Overview of the 17 testbeds which are federated as part of Fed4FIRE

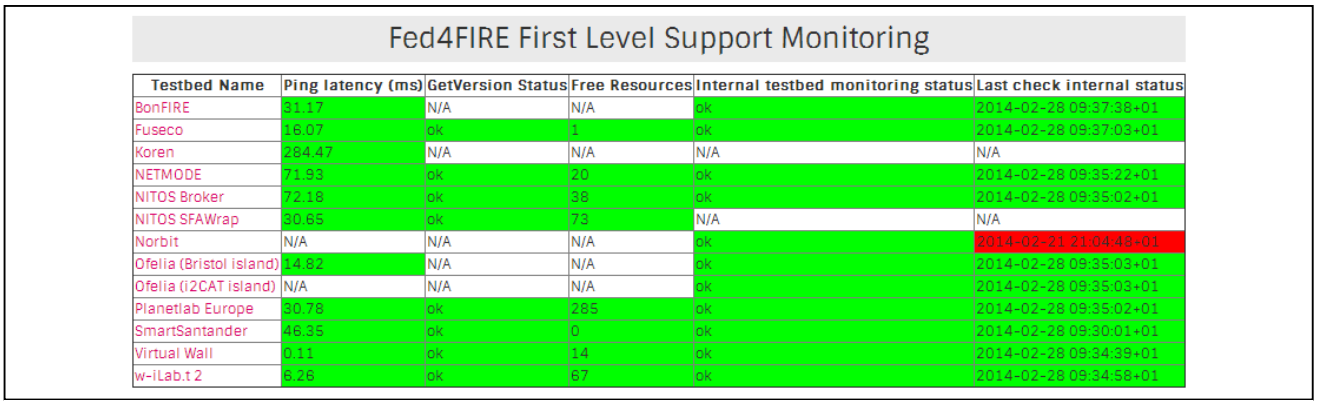
The remainder of the paper is structured as follows: In Section II we describe which design considerations were made to create the health monitoring system. In Section III we outline the practical implementation in Fed4FIRE. We then make an evaluation of the current setup in Section IV. We conclude with an outlook on the future work in Section V and a summary in section VI.

II. Design Considerations

It is not straightforward to determine the indicators that should be included in the health status information of the federated testbeds, or how they should be presented to the operators or experimenters. Experience gained from a similar federated operational environment in the sector of backbone networks made clear that the status information should be presented as concisely as possible. The form of a dashboard is deemed to be the most suitable for this. Typically operational dashboards indicate the health status of the monitored components using a Red/Amber/Green (RAG) color scheme. Adopting the same approach for the federation's health status overview seems appropriate.

Then the question remains: which information must be captured and displayed on this RAG dashboard? The example of the pan-European research network described in the introduction can be a useful source of inspiration. The health monitoring of that federated network of NRENs and the GÉANT interconnecting network is based on two different types of input: measurements coming from the NRENs themselves, and measurements specifically performed by the operator of the federated network. In the context of Fed4FIRE, this model would translate to a fusion of health monitoring data provided by the testbeds with specific federation-level health monitoring data.

Based on the existing internal status monitors of the participating testbeds in Fed4FIRE, several possible indicators were investigated. In the end, five key indicators were identified which together give an overview of the status of each testbed. The indicators are spread across multiple layers of functionality. In case of a malfunction, this allows for a quick identification of where the problem is situated. The first three indicators are monitored by the federation: on the lowest level, network connectivity to the testbed is checked. Secondly, a simple query is performed to the testbed's management server to check if it is up and running. The third indicator is the available resources present on the testbed. When no resources are available, swapping in a new experiment will fail. It is important for experimenters and FLS to be aware of this number, as making requests that exceed this resource limit is one of the most common reasons for an experiment to fail. The fourth key indicator is provided by the testbeds themselves: they must provide the centralized health monitor with an aggregated Red/Amber/Green internal status, which is directly displayed on the dashboard. Each testbed has the freedom to define for itself when its infrastructure should be considered to be in which of those three states. This can differ from testbed to testbed. The fifth indicator is the timestamp of when this aggregated internal status was refreshed for the last time. When this timestamp becomes too old, the information provided by



Fed4FIRE First Level Support Monitoring

Testbed Name	Ping latency (ms)	GetVersion Status	Free Resources	Internal testbed monitoring status	Last check internal status
BonFIRE	31.17	N/A	N/A	ok	2014-02-28 09:37:38+01
Fuseco	16.07	ok	1	ok	2014-02-28 09:37:03+01
Koren	284.47	N/A	N/A	N/A	N/A
NETMODE	71.93	ok	20	ok	2014-02-28 09:35:22+01
NITOS Broker	72.18	ok	38	ok	2014-02-28 09:35:02+01
NITOS SFAWrap	30.65	ok	73	N/A	N/A
Norbit	N/A	N/A	N/A	ok	2014-02-21 21:04:48+01
Ofelia (Bristol island)	14.82	N/A	N/A	ok	2014-02-28 09:35:03+01
Ofelia (i2CAT island)	N/A	N/A	N/A	ok	2014-02-28 09:35:03+01
Planetlab Europe	30.78	ok	285	ok	2014-02-28 09:35:02+01
SmartSantander	46.35	ok	0	ok	2014-02-28 09:30:01+01
Virtual Wall	0.11	ok	14	ok	2014-02-28 09:34:39+01
w-iLab.t 2	5.26	ok	67	ok	2014-02-28 09:34:58+01

Fig. 2. Continuous Monitoring Dashboard as used by FLS within Fed4FIRE

the testbed must be considered out-of-date, and typically indicates a malfunction of the testbed infrastructure.

These five key indicators can together give a rather complete, composite overview of the health status of the federated testbeds, but an even more thorough indication of the testbeds' health can be attained by performing *automated scenario tests* which go through all the steps of the experimental lifecycle. As such tests require significantly more resources than continuous passive monitoring, the frequency of testing is much lower and they should only be executed during off-peak periods.

III. Implementation in Fed4FIRE

In Fed4FIRE, the health monitoring system currently consists of two components: a dashboard which continuously monitors the five identified key indicators, and an information panel with the latest status of each *automated scenario test* that is run within the federation. To limit the resource usage of these tests, they are run two times a day: once in the morning, and once in the evening, as these prove to be off-peak hours. However, when needed, an FLS operator can manually initiate an extra test run to check the status of the federation.

A. Continous monitoring dashboard

The dashboard that is currently in use in Fed4FIRE can be seen in figure 2. It is a service provided by the federation, and currently being maintained by iMinds. Each row contains the key indicators of a testbed, which are visually enriched by an automatically computed Red/Amber/Green (RAG) status. This allows operators to easily spot problematic values.

An overview of the implementation of the health monitoring system is given in figure 3. Centrally within the system, all information is stored into a PostgreSQL-database. This database can be queried by the First Level Support dashboard, but can also be used for computing long-term statistics, generating e-mail alarms, etc.

The information in this database is gathered by different components. The first component performs network connectivity tests to a testbed with an ICMP Echo-request (*ping*) to the *Aggregate Manager* (AM), which is the central management server of the testbed. This is the main server that is contacted for all requests originating from the federation to that testbed. When pinging this server fails, a general networking failure is the most likely cause. By monitoring the number of outages and the evolution of these round trip times, the quality of the connection to the testbed can also be evaluated.

To fetch the second and third indicator, jFed[4] is used. This is a Java-based framework developed by iMinds which supports all API's within the Fed4FIRE federation. It contains automated testing tools, a probe for manual testing and an end-user experimentation-toolkit. By using this framework for health monitoring and end-user usage alike, we can verify and guarantee the correct functioning of the federation. For the second key indicator, the jFed probe performs a basic information call (*GetVersion*) to the API of the testbed's Aggregate Manager (AM). This call returns basic information about the testbed, such as supported protocols and internal software version information. A correct answer to this query is used as an indicator of whether the AM is up and responsive. One of the most useful parameters returned in this call is an identifier (codetag) of the testbed's internal software version. When a problem arises, this identifier can be used to check whether the defect was introduced by an update of the testbed's software. The third indicator is obtained by a *ListResources*-call to the AM. From the result of this call, the number of available resources is calculated. These two indicators can also be used to verify that the federated authentication mechanism is functioning correctly. The *ListResources*-call requires valid authentication credentials, while these are not needed for the *GetVersion*-call. If jFed can successfully make the *GetVersion*-call, but fails on the *ListResources*-call, this will typically indicate an authentication issue.

The fourth indicator, the internal status of the testbed, is monitored by the facility monitoring of the testbed itself. Each testbed is expected to summarize its status into a Red/Amber/Green state, and push this status via an OML-stream (Orbit Measurement Library-stream) [5] to an OML-server of the federated health monitoring system. The testbed may also include additional information about the status of the different parts of the testbed's system. This information can be different per testbed. For instance, the Virtual Wall testbed of iMinds makes use of specific VLAN-aware Ethernet switches to automatically emulate any desired resource topology.

Therefore health information about these switches (on/off, network load per port, etc.) can be considered to be important when determining the health status of that testbed. While for a wireless testbed it might be more interesting to capture if there is too much wireless interference. But, independent of the actual captured data, providing information about the status of the different parts of the testbed's system allows the testbed's operational staff to give FLS and experimenters more insight into the current status of the testbed. All information pushed to the OML-server is captured in an OML-specific PostgreSQL database. From there it is processed and the aggregated RAG-state, together with the timestamp of when the information was received, are inserted into the central health-monitoring system database. This gives us the fourth and fifth key indicator. Note that as mentioned before, every testbed can decide autonomously how its aggregate status has to be calculated.

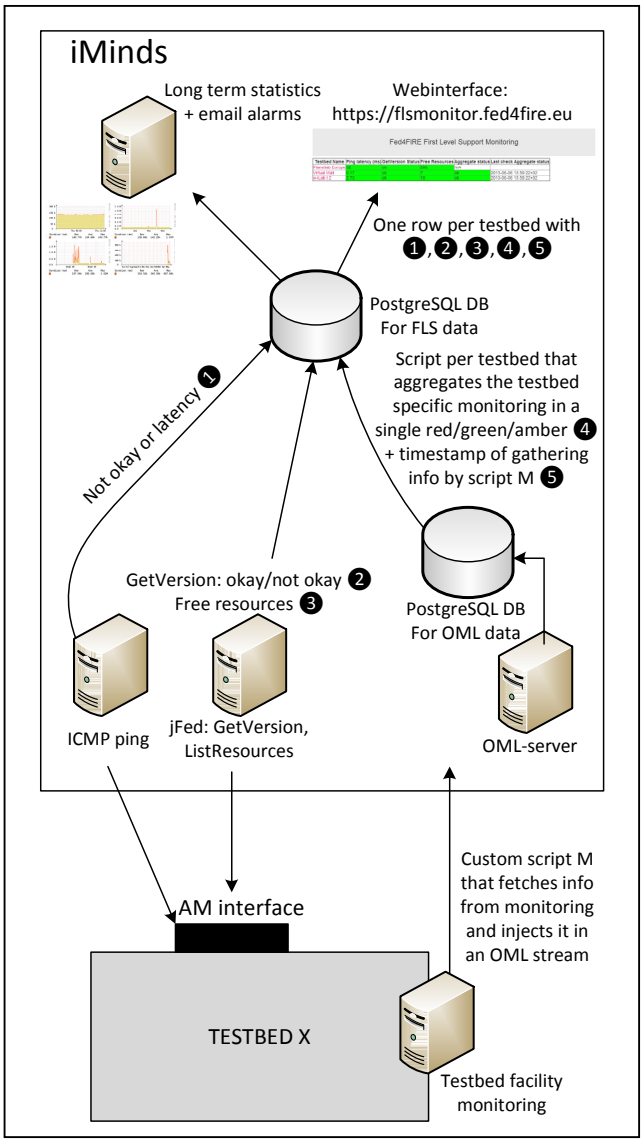


Fig. 3. Current Health monitoring infrastructure within Fed4FIRE

B. *Automated scenario tests*

jFed is used to perform *automated scenario tests*. These scenarios perform all steps that are typically performed during an experiment. For each of these steps, all requests made and responses received are logged into a database. Each response is verified on several levels. First the response is checked against the protocol standards, and discrepancies are logged. Secondly, the response is matched against the expected outcome. In the case where the testbed replied with an error message, this check can be skipped. Finally, all necessary information is extracted to be able to perform the subsequent calls and/or check the outcome of those calls. All irregularities are logged according to their varying degrees of severity. In case of a fatal error, all dependent successive steps are discarded.

The most important automated scenario tests that are currently in use perform setup-and-login tests on each testbed. In case of Fed4FIRE, which is designed around the Slice-based Federation Architecture (SFA), the test suite first acquires user credentials with sufficient rights to perform actions on the tested facility, and creates a new *slice* on the central *Slice Authority* (SA). Then the following steps are performed on the testbed under review:

1. Request the supported protocols of the testbed's *Aggregate Manager* (AM), and verify that all required protocols are listed.
2. Allocate a sliver at the AM with one node.
3. Provision the sliver on the testbed.
4. Request the status of the sliver until the testbed indicates the sliver is swapped in.
5. Request a description of the sliver.
6. Log in onto the requested node through SSH, and request some basic system information.
7. Remove the sliver from the testbed.

Another type of *automated scenario tests* that is currently in use within Fed4FIRE tests the setup of experiments which construct layer 2-links between nodes on different testbeds. To achieve this, multiple VLAN's on research networks such as those employing AutoBAHN [6] and Internet2's ION [7] are stitched together to achieve the layer 2-connectivity. In this test, for each combination of testbeds that support stitching the following steps are executed after creating a new *slice* on the central SA:

1. Call the Stitching Computation Service (SCS) to receive instructions on how to setup a layer 2-link between the two testbeds.
2. Allocate and provision a sliver with one node on each of the two testbeds.
3. Follow the instructions of the SCS to configure the layer 2-link.
4. Log in onto both nodes through SSH, and try to ping the other party.
5. Remove the slivers from the testbed and destroy the layer 2-link.

IV. EVALUATION

The Fed4FIRE project includes three subsequent development cycles. The first of them ended on January 31st 2014, meaning that the Fed4FIRE federation went operation for the first time on February 1st 2014. In order to prepare for this important milestone, the FLS services started operation on 6th January 2014 and operated in a pilot mode for the remainder of that month so that the operations staff could gain 'live' experience of operating the service. For this they could rely on the health monitoring dashboard which is described in section III, a Trouble Ticket Service (TTS) based on the JIRA issue tracking software, and a shared calendar for the announcement of testbed maintenance. During the period January 6-31st, 25 trouble tickets were opened. 23 of these were as a result of dashboard alarms detected by the FLS staff. There was one scheduled maintenance during the period. This illustrates that the federation's health dashboard and underlying framework are essential components for the federation's First Level Support service to guard the operational status of the entire federation. The 5 key indicators collected in the FLS dashboard, together with the more detailed information of the automated SSH login tests turned out to be providing sufficient health information. The use of the dashboard by FLS staff highlighted certain limitations relating to the way information is displayed, access to subsidiary information and ease-of-use to follow up alarms. These have been discussed with iMinds and a number of improvements are planned. Amongst other improvements, it will be made possible to query historical data, which will allow the FLS to gather trend information. Supplementary information will be made available, which will allow FLS operators to diagnose more accurately why a testbed is failing. To improve practical usability, FLS operators will be given the ability to post comments on the dashboard. Besides the technical remarks, the pilot also made clear that the operational processes of each testbed, such as the performance of urgent maintenance, need to be enhanced to notify the federation in order to prevent unnecessary action on alarms by the FLS.

V. FUTURE WORK

Currently, each testbed has its own infrastructure monitoring framework, and has complete liberty over how it provides information about the internal status of parts of the testbed. Efforts are ongoing to standardize which minimum information must be included in this breakdown.

The feedback of the FLS operators is currently being processed. The dashboard is currently being enhanced to give access to the historical values of the key indicators.

Later on, this historical data will be compiled into performance and reliability statistics that will be used to enforce SLA's within the Fed4FIRE federation. These SLA's are currently being drafted, and will also provide input for potential extra indicators that must be monitored.

Furthermore, these statistics will be used to enhance end-user tools of the jFed toolkit. By giving experimenters access to these statistics, they will be able to make an informed decision on which components suite their needs best. Moreover, the jFed toolkit will also be enhanced to collect metrics while actual experiments are being performed by the experimenters. Besides all these automatically gathered metrics, perceived performance and reliability will also be included into the metrics. This will be achieved by asking the end-user score his experiment after its conclusion. This extra information will then be used to further improve the formula with which the performance and reliability statistics are being computed.

VI. CONCLUSIONS

The operational management of a federation of Future Internet Research Facilities is a challenging undertaking. This paper outlined how a centralized health monitoring system has the potential to ease the life of experimenters and facility providers alike. We analyzed which monitoring information was needed, and how it could be displayed in an effective fashion. This health monitoring system was then developed on the jFed framework, and is now running in production. After one month of usage, we learned that our health monitoring system is an important instrument to detect anomalies and improve the allover reliability within the federation. Only a few minor problems with the dashboard and underlying framework could be determined, and these are being addressed as soon as possible.

ACKNOWLEDGMENT

This work was carried out with the support of the Fed4FIRE project ("Federation for FIRE"), an integrated project funded by the European Commission through the 7th ICT Framework Programme (318389). It does not necessarily reflect the views of the European Commission. The European Commission is not liable for any use that may be made of the information contained herein.

REFERENCES

[1] Fed4FIRE. (2014, February). European Funded Project Fed4FIRE – Federation for FIRE [Online] Available: http://www.fed4fire.eu/

[2] Vandenberghe, Wim, et al. "Architecture for the heterogeneous federation of future internet experimentation facilities." in *Future Network and Mobile Summit (FNMS), 2013*. IEEE, 2013.

[3] Davies, Dai "Network Reliability – the Role of Excellence In Network Operations" in *Design of Reliable Communication Networks (DRCN), 2014*. IEEE, 2014.

[4] iMinds (2014, February). jFed: Java based framework to support SFA testbed federation client tools [Online] Available: http://jfed.iminds.be/

[5] M. Singh, M. Ott, I. Seskar, and P. Kamat. "Orbit measurements framework and library (oml): motivations, implementation and features," in *Proceedings of the 1st International Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities, 2005*. Tridentcom 2005. 146-152.

[6] M. Campanella, R. Krzywania et al, "Bandwidth on Demand Services for European Research and Education Networks" in *Proceeding of the 1st IEEE International Workshop on Bandwidth on Demand*, Nov. 2006, San Francisco, USA, Volume 1, Nov. 2006. 65 – 72.

[7] Sterbenz, James PG, et al. "The great plains environment for network innovation (GpENI): a programmable testbed for future internet architecture research."*Testbeds and Research Infrastructures. Development of Networks and Communities*. Springer Berlin Heidelberg, 2011. 428-441.